Consensus Decision Trees: Using Consensus Hierarchical Clustering for Data Relabelling and Reduction

Branko Kavšek[1], Nada Lavrač[1], and Anuška Ferligoj[2]

[1] Institute Jožef Stefan, Jamova 39, 1000 Ljubljana, Slovenia
branko.kavsek@ijs.si, nada.lavrac@ijs.si
[2] University of Ljubljana, 1000 Ljubljana, Slovenia
anuska.ferligoj@uni-lj.si

Abstract. In data analysis, induction of decision trees serves two main goals: first, induced decision trees can be used for classification/prediction of new instances, and second, they represent an easy-to-interpret model of the problem domain that can be used for explanation. The accuracy of the induced classifier is usually estimated using N-fold cross validation, whereas for explanation purposes a decision tree induced from all the available data is used. Decision tree learning is relatively non-robust: a small change in the training set may significantly change the structure of the induced decision tree. This paper presents a decision tree construction method in which the domain model is constructed by consensus clustering of N decision trees induced in N-fold cross-validation. Experimental results show that consensus decision trees are simpler than C4.5 decision trees, indicating that they may be a more stable approximation of the intended domain model than decision tree, constructed from the entire set of training instances.

1 Introduction

Decision tree induction (Breiman et al. 1984, Quinlan, 1986) has been recognized as one of the standard data analysis methods. In particular, variants of Quinlan's C4.5 (Quinlan, 1993) can be found in virtually all commercial and academic data mining packages.

In data analysis, induction of decision trees serves two main goals: first, induced decision trees can be used for the classification (or prediction) of new instances, and second, they represent an easy-to-interpret model of the problem domain that can be used for explanation. In the standard decision tree learning methodology (e.g., as implemented in the WEKA system (Witten & Frank, 1999)) the accuracy of the induced classifier is estimated using N-fold cross-validation, whereas for explanation purposes a decision tree induced from all the available data is used. Its explanation capability is evaluated qualitatively by the domain expert, whereas quantitative measures estimate only the simplicity of decision trees, measured by the number of leaves and nodes.

L. De Raedt and P. Flach (Eds.): ECML 2001, LNAI 2167, pp. 251–262, 2001.

The main advantages of decision tree learning are computational efficiency, reasonable accuracy and simplicity of explanations. It is well known, however, that decision tree learning is a rather non-robust method: a small change in the training set may significantly change the structure of the induced decision tree, which may result in experts' distrust in induced domain models. Improved robustness and improved accuracy results can be achieved e.g., by bagging/boosting (Breiman, 1996) at a cost of increased model complexity and decreased explanatory potential.

This paper addresses the model selection problem of the standard decision tree learning methodology in which the induced domain model is the decision tree induced from all the available data. The accuracy of the induced classifier is estimated using N-fold cross-validation, which is a bias-free (Stone, 1974) but not variance-free (Zhang, 1992, Kohavi, 1995) estimate of the true accuracy, i.e., the accuracy of a classifier that is learned by the same algorithm on the complete data set. For model selection purposes, a 63,2% bootstrap (Efron 1979) may be preferable to learning from a complete set of training instances (Scheffer & Herbrich, 1997). The bootstrap approach is based on re-sampling of a number of training sets of size n from an original data set of size n by randomly drawing samples with replacement, leading to 63,2% distinct samples in the training set, on the average.

Despite the statistical advantages of this method for choosing the optimal model, this method is still non-robust in the case of decision tree learning. To improve robustness, this paper presents a decision tree construction method in which the domain model in the form of a decision tree is constructed by consensus clustering of N decision trees induced in N-fold cross-validation. Experimental results show that consensus decision trees are simpler than C4.5 decision trees, indicating that they may be a more stable approximation of the intended domain model than decision trees constructed from the entire set of training instances.

The paper is organized as follows. Section 2 presents the basic methodology of decision tree induction and hierarchical clustering, Section 3 outlines the novel approach of consensus decision tree construction, and Section 4 provides the experimental evaluation of the proposed approach. We conclude by a summary and plans for further work.

2 Background Methodology

2.1 Decision Trees

Induction of decision trees is one of the most popular machine learning methods for learning of attribute-value descriptions (Breiman et al., 1984, Quinlan, 1986). The basic decision tree learning algorithm builds a tree in a top-down greedy fashion by recursively selecting the 'best' attribute on the basis of an information measure, and splitting the data set accordingly. Various modifications of the basic algorithm can be found in the literature, the most popular being Quinlan's C4.5 (Quinlan, 1993). In our work we used the WEKA (Witten and Frank, 1999) implementation of C4.5.

2.2 Hierarchical Clustering

Clustering methods in general aim at building clusters (groups) of objects so that similar objects fall into the same cluster (internal cohesivity) while dissimilar objects fall into separate clusters (external isolation). A particular class of clustering methods, studied and widely used in statistical data analysis (e.g., Sokal and Sneath, 1963; Gordon, 1981; Hartigan, 1975) are *hierarchical clustering* methods.

The purpose of hierarchical clustering is to fuse objects (instances) into successively larger clusters, using some measure of (dis)similarity. A typical result of this type of clustering is a hierarchical tree or dendrogram.

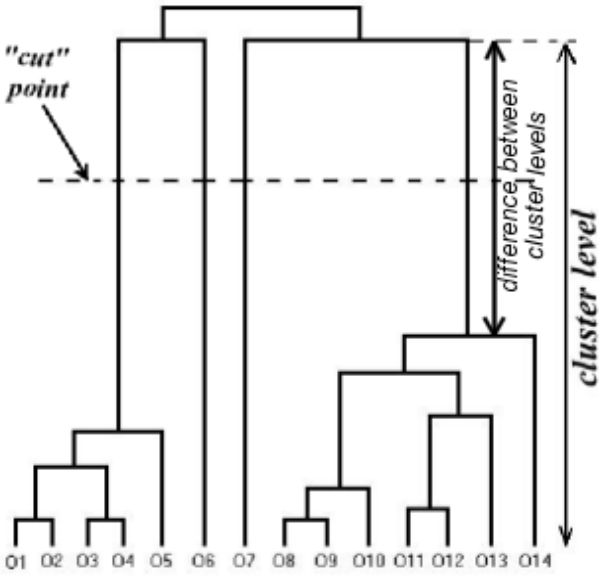

Fig. 1. A sample dendrogram obtained as a result of hierarchical clustering.

As shown in Figure 1, a dendrogram is a binary tree where single objects form the leaves of the tree and each node of the tree represents a cluster of similar objects. The further the node is from the tree root, the more similar the items are under the node. The height of the branches (vertical lines) in a dendrogram are directly proportional to the dissimilarity between clusters. Thus, for each node in the dendrogram (where a new cluster is formed) we can read off the dissimilarity at which the respective objects were joined together into a new single cluster. This dissimilarity is called the *cluster level* and is used to determine the most appropriate number of clusters that reflects the real structure in the data.

The dendrogram illustrates the actual procedure of hierarchical clustering. It starts by forming N clusters, each consisting of one single object (training instance). Then, step by step, the threshold regarding the decision when to

declare two objects to be members of the same cluster is lowered. As a result, larger clusters of increasingly dissimilar objects are aggregated. Finally, in the last step, all objects are joined to form a single cluster. The cluster levels are computed and where the difference between successive cluster levels is maximal (Figure 1) the dendrogram is 'cut', producing the partition where each cluster is the most internally cohesive and there is the highest external isolation between clusters. Note that the number of clusters is determined dynamically through the procedure of dendrogram 'cutting'.

There is one last question that remains to be answered in order to understand the hierarchical clustering: "how should we measure the (dis)similarity between objects and between clusters of objects?" These questions are addressed in the following two paragraphs, respectively.

Dissimilarity measures. The hierarchical clustering method uses the dissimilarities between objects when forming the clusters. The most straightforward way of computing dissimilarities between objects in a multi-dimensional space is to compute the Euclidean distances; many other dissimilarity measures are also used in clustering algorithms (e.g., Gordon, 1981: 13-32). In our CDT algorithm, described in Section 3, we use a modified disagreement dissimilarity measure which we describe in detail in Section 1.

Aggregation or linkage rules. At the first step, when each object represents its own cluster, the dissimilarities between these objects are defined by the chosen dissimilarity measure. Once several objects have been aggregated, the dissimilarities between these new clusters has to be determined. There are various possibilities: for example, the dissimilarity between the fused cluster $(C_i \cup C_j)$ and another cluster (C_k) can be the smallest dissimilarity between $d(C_i, C_k)$ and $d(C_j, C_k)$; this method is called the *single linkage* method. Alternatively, one may use the largest dissimilarity between $d(C_i, C_k)$ and $d(C_j, C_k)$, i.e., $d(C_i \cup C_j, C_k) = max(d(C_i, C_k), d(C_j, C_k))$. This method, called the *complete linkage* method, has been used in our CDT algorithm. There are numerous other linkage rules (e.g., Gordon, 1981).

An example of hierarchical clustering. To better illustrate the hierarchical clustering method, a simple example is presented in Figure 2. Having five points (x,y,z,w,v) in a two-dimensional space (Figure 2a), we want to assign these points to clusters. Taking the Euclidian distance between points as the dissimilarity measure, we compute the dissimilarity matrix (Figure 2b). What follows is one step of the hierarchical clustering algorithm: find the smallest dissimilarity value in the matrix (Figure 2b - encircled value), fuse the appropriate elements together (Figure 2b - points x and y), delete from the matrix the row and column containing this (smallest) value and recompute the dissimilarity matrix according to the complete linkage aggregation rule. Figure 2c is what we get after applying one step of this algorithm. Repeating the step until the dissimilarity matrix 'shrinks' to a single value (Figures 2c,d,e; the last step is not shown), we obtain a dendrogram (Figure 2f). The cluster levels 1, 1, 1.41

and 5.66 correspond to the encircled values in Figures 2b,c,d,e; the differences between successive cluster levels are thus: 0, 0.41 and 4.25. The dendrogram in Figure 2f is cut where this difference is maximal (4.25), yielding two clusters of points: (x,y,z) and (w,v).

Consensus clustering. Consensus hierarchical clustering deals with the following problem: given a set of concept hierarchies (represented by dendrograms), find a *consensus concept hierarchy* by merging the given concept hierarchies in such a way that similar instances (those that belong to the same concept/cluster) will remain similar also in the merged concept hierarchy. In the last thirty years many consensus clustering methods have been proposed (e.g., Regnier, 1965; Adams, 1972; McMorris and Neuman, 1983; Day, 1983). In 1986, a special issue of the Journal of Classification was devoted to consensus classifications. Excellent reviews of this topic are also available (Faith, 1988; Leclerc, 1988).

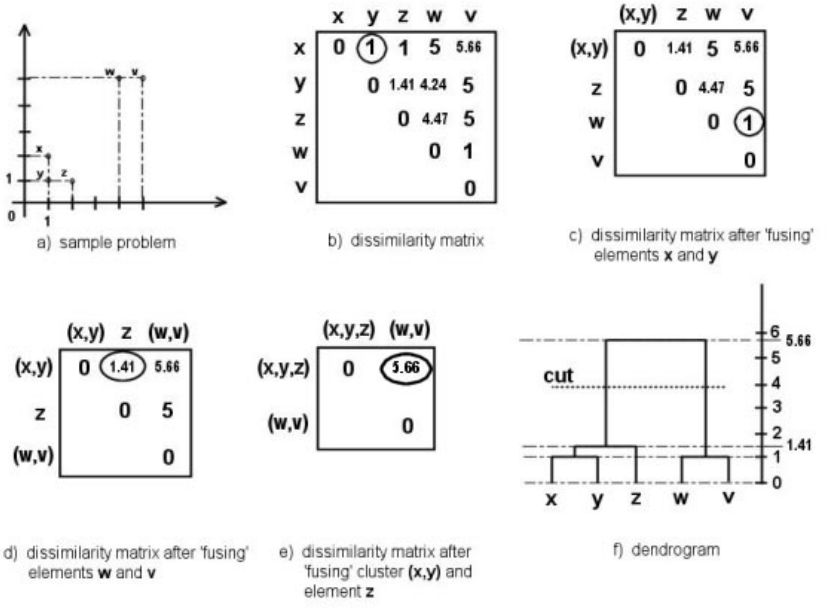

Fig. 2. A short example problem solved using hierarchical clustering with complete linkage aggregation rule.

3 Consensus Decision Tree Construction

3.1 Motivation

As pointed out by Langley (Langley, 1996), decision tree induction can be seen as a special case of induction of concept hierarchies. A concept is associated with

each node of the decision tree, and as such a tree represents a kind of taxonomy, a hierarchy of many concepts. In this case, a concept is identified by the set of instances in a node of the decision tree. Hierarchical clustering also results in a taxonomy of concepts, equally identified by the set of instances in a 'node' of a dendrogram representing the concept hierarchy. Concept hierarchies can be induced in a supervised or unsupervised manner: decision tree induction algorithms perform supervised learning, whereas induction by hierarchical clustering is unsupervised.

Our idea of building consensus decision trees is inspired by the idea of consensus hierarchical clustering. A consensus decision tree should be constructed in such a way that instances that are similar in the original decision trees should remain being similar also in the consensus decision tree. To this end, it is crucial to define an appropriate measure of similarity between instances. This measure may only consider the distances between instances or may also profit from the fact that instances are labelled by class labels and appropriately increase the similarity value of two instances labelled by the same class label. We have tested both approaches.

3.2 CDT: An Algorithm for Consensus Decision Tree Construction

The consensus tree building procedure consists of the following main steps:

1. perform N-fold cross-validation resulting in N decision trees induced by a decision tree learning algorithm (e.g., C4.5),
2. use these decision trees for computing a dissimilarity matrix that measures the dissimilarity of pairs of instances,
3. construct a concept hierarchy using the dissimilarity matrix of step 2 and define concepts by 'cutting' the dendrogram w.r.t. the maximal difference in cluster levels,
4. induce a consensus decision tree using the same decision tree algorithm as in step 1.

Decision tree construction. First, N-fold cross-validation is performed resulting in N decision trees induced by the C4.5 learning algorithm (the WEKA implementation of C4.5 with default parameters[1] is used for this purpose). The decision trees are then stored and used to compute the dissimilarity between pairs of instances.

Dissimilarity between instances. The dissimilarities between pairs of instances are computed from the N stored decision trees in the following way:

– first we measure the similarity s between instances i and j by counting (for all N decision trees) how many times the two instances belong to the same leaf (i.e., are described by the same path of attribute-value tests leading

[1] The default parameters were: *binary splits = NO*, *confidence factor for pruning = 0.25*, *minimum number of objects in a leaf = 2*.

from the root to the leaf of the decision tree). Therefore $s(i,j)$ is defined as follows:

$$s(i,j) = \sum_{l=1}^{N} T_l(i,j)$$

where

$$T_l(i,j) = \begin{cases} 1, \text{ if } i \text{ and } j \text{ belong to the same leaf (are described by} \\ \quad \text{the same attribute values) in the } l\text{−th decision tree} \\ 0, \text{ otherwise} \end{cases}$$

– then we compute the dissimilarity measure $d(i,j)$ by simply subtracting the similarity $s(i,j)$ from the number of trees N, i.e., $d(i,j) = N - s(i,j)$, which gives the same results as its normalized variant:

$$d(i,j) = 1 - \frac{s(i,j)}{N}$$

By calculating the dissimilarities for all pairs of instances, we obtain the dissimilarity matrix, which is the input for the hierarchical clustering algorithm.

Concept hierarchy construction. A concept hierarchy is constructed using the following hierarchical clustering algorithm:

Each instance is a cluster: $C_i = \{i\}$;
REPEAT
    find the 'nearest' pair of clusters C_p and C_q:
        $d(C_p, C_q) = \min_{u,v} d(C_u, C_v)$;
    fuse clusters C_p and C_q in a new cluster $C_r = C_p \cup C_q$;
    replace clusters C_p and C_q by the cluster C_r;
    determine the dissimilarities between C_r and the other clusters
        using the complete linkage method;
UNTIL one cluster is left

This algorithm produces a dendrogram as its output. The concepts (clusters) can then be identified by 'cutting' this dendrogram according to the maximal difference in cluster levels (as described in Section 2.2). If needed, we increase the height of the cutpoint to ensure that the number of clusters remains greater or equal to the number of classes of the given classification problem. Consequently, we sometimes force the algorithm to cut the dendrogram producing more clusters than the optimal number of clusters according to the maximal difference between successive cluster levels.

Induction of consensus decision trees. Within each cluster of instances, we select the majority class. We have tested the following versions of the algorithm:

Learning by data relabelling. In this algorithm, not reported in this paper, we re-classify the instances belonging to non-majority classes by assigning them to the majority class. We then use the C4.5 learning algorithm to induce a consensus decision tree from the set of all instances, some being relabelled.

Learning by data reduction. In the algorithm reported in this paper, we remove from the cluster all instances not belonging to the majority class. We then use the C4.5 learning algorithm to induce the consensus decision tree from the remaining subset of instances.

In all runs of the C4.5 algorithm the same (default) parameter setting is used (as in the first step of this algorithm). Notice that in the case of a tie (two or more classes being the majority class), a random choice between these class assignments is made. The results of learning by data reduction slightly (non-significantly) outperform the results of learning by data relabelling. Due to the lack of space, only the results of learning by data reduction are reported in Table 2.

4 Experimental Evaluation

4.1 Experimental Design

In standard 10-fold cross-validation, the original data set is partitioned into 10 folds with (approximately) the same number of examples. Training sets are built from 9 folds, leaving one fold as a test set. Let G denote the entire data set, T_i an individual test set (consisting of one fold), and G_i the corresponding training set ($G_i \leftarrow G \setminus T_i$, composed of nine folds). In this way, 10 training sets G_1– G_{10}, and 10 corresponding test sets T_1–T_{10} are constructed. Every example occurs exactly once in a test set, and 9 times in training sets.

In the first experiment we used C4.5 (WEKA implementation, default parameter setting[2]) to induce decision trees on training sets G_1–G_{10}. We measured the average accuracy $Acc(C4.5(G))$ (and standard deviation) and the information score[3] $Info(C4.5(G))$ of ten hypotheses $C4.5(G_i)$ constructed by C4.5 on training sets G_i, $i \in [1, 10]$, where $Acc(C4.5(G)) = \frac{1}{10}\sum_1^{10} Acc(C4.5(G_i))$, and $Info(C4.5(G)) = \frac{1}{10}\sum_1^{10} Info(C4.5(G_i))$. The average size of decision trees $Leaves/Nodes(C4.5(G))$ was measured by the number of leaves and the number of all decision tree nodes (number of leaves + number of internal nodes), averaged over 10 folds.

The above result presents the baseline for comparing the quality of our consensus tree building algorithm CDT, measured by the average accuracy $Acc(CDT(G))$, $Leaves/Nodes(CDT(G))$, and information score $Info(CDT(G))$ over ten consensus decision trees $CDT(G_i)$.

[2] See footnote 1 in Section 3.2.

[3] Whereas accuracy computes the relative frequency of correctly classified instances, the information score takes into the account also the improvement of accuracy compared to the prior probability of classes, see (Kononenko and Bratko, 1991).

As described in Section 3.2, a consensus decision tree is constructed from ten C4.5 decision trees. Building of consensus decision trees was performed in a nested 10-fold cross-validation loop: for each G_i, $i \in [1, 10]$, training sets G_{ij}, $j \in [1, 10]$ were used to construct decision trees $C4.5(G_{ij})$ by the C4.5 algorithm. Training sets G_{ij} were obtained by splitting each G_i into ten test sets T_{ij} (consisting of one sub-fold), and ten training sets $G_{ij} \leftarrow G_i \setminus T_{ij}$ (composed of nine sub-folds).

Ten decision trees $C4.5(G_{ij})$ were merged into a single consensus decision tree $CDT(G_i)$. Let $Acc(CDT(G_i))$ denote its accuracy tested on T_i. Accordingly, $Acc(CDT(G)) = \frac{1}{10}\sum_1^{10} Acc(CDT(G_i))$, and information contents $Info(CDT(G)) = \frac{1}{10}\sum_1^{10} Info(CDT(G_i))$. $Leaves/Nodes(CDT(G))$ is also the average of $Leaves/Nodes(CDT(G_i))$.

In order to compare accuracy, tree size and information score of consensus trees and C4.5 trees, we calculate their *relative improvements* as follows: $Rel(Acc(G)) = \frac{Acc(CDT(G))}{Acc(C4.5(G))} - 1$, $Rel(Leaves/Nodes(G)) = 1 - \frac{Leaves/Nodes(CDT(G))}{Leaves/Nodes(C4.5(G))}$, $Rel(Info(G)) = \frac{Info(CDT(G))}{Info(C4.5(G))} - 1$.

Table 1. Characteristics of data sets.

Data set	#Attr.	#Class.	#Inst.	Class distribution (%)
Anneal	38	5	898	1:11:**76**:8:4
Audiology	69	24	226	1:1:**25**:9:1:1:8:21:1:2:1:2:1:1:3:1:1:1:9:2:1:2:4:1
Australian	14	2	690	**56**:44
Autos	25	6	205	1:11:**33**:26:16:13
Balance	4	3	625	**46**:8:**46**
Breast	9	2	286	**70**:30
Breast-w	9	2	699	**66**:34
Car	6	4	1728	**70**:22:4:4
Colic	22	2	368	**63**:37
Credit-a	15	2	690	44:**56**
Credit-g	20	2	1000	**70**:30
Diabetes	8	2	768	**65**:35
Glass	9	6	214	33:**36**:8:6:4:14
Heart-c	13	2	303	**54**:46
Heart-stat	13	2	270	**56**:44
Hepatitis	19	2	155	21:**79**
Ionosphere	34	2	351	36:**64**
Iris	4	3	150	**33**:**33**:**33**
Labor	16	2	57	35:**65**
Lymph	18	4	148	1:**55**:41:3
Prim. tumor	17	21	339	**25**:5:3:4:11:1:4:2:1:8:4:2:7:1:1:3:8:2:1:1:7
Segment	19	7	2310	**14**:**14**:**14**:**14**:**14**:**14**:**14**
Sonar	60	2	208	47:**53**
Tic-tac-toe	9	2	958	**65**:35
Vehicle	18	4	846	25:**26**:**26**:24
Vote	16	2	435	**61**:39
Wine	13	3	178	33:**40**:27
Zoo	17	7	101	**41**:20:5:13:4:8:10

4.2 Results of Experiments

Experiments were performed on 28 UCI data sets whose characteristics are outlined in Table 1 (boldface denoting the majority class). Results of experiments are shown in Table 2 (boldface meaning that CDT performed equally well or better than C4.5).

Table 2. Results of the experiments.

Data set	C4.5			CDT			Relative improvement		
	Acc(Sd)	leaves /nodes	info.	*Acc(Sd)*	leaves /nodes	info.	Acc	leaves /nodes	info.
Anneal	97,14 (0,0893)	39/77	2,7130	**99,13** (0,0497)	52/103	**2,7811**	**0,0205**	-0,333/-0,338	**0,0251**
Audiol.	77,88 (0,1193)	32/54	2,6579	**80,09** (0,1288)	**29/49**	2,5316	**0,0284**	**0,094/0,093**	-0,0475
Austral.	85,51 (0,3455)	31/45	0,6183	84,64 (0,3919)	**17/24**	**0,6813**	-0,0102	**0,452/0,467**	**0,1019**
Autos	82,44 (0,2022)	49/69	1,8198	80,49 (0,2361)	**39/57**	1,7598	-0,0237	**0,204/0,174**	-0,0330
Balance	77,76 (0,3567)	58/115	0,6778	69,12 (0,4537)	**8/15**	0,5598	-0,1111	**0,862/0,870**	-0,1741
Breast	75,17 (0,4423)	4/6	0,1005	72,73 (0,5222)	13/17	**0,2630**	-0,0326	-2,250/-1,833	**1,6169**
Brst-w	95,28 (0,2116)	16/31	0,8020	94,85 (0,2269)	**9/17**	**0,8189**	-0,0045	**0,438/0,452**	**0,0211**
Car	92,48 (0,1628)	131/182	1,0218	68,66 (0,3946)	**29/41**	0,2687	-0,2576	**0,779/0,775**	-0,7370
Colic	85,87 (0,3518)	4/6	0,4993	83,42 (0,4071)	11/17	**0,6023**	-0,0285	-1,750/-1,833	**0,2063**
Crdt-a	85,94 (0,3402)	30/42	0,6221	85,07 (0,3864)	**18/25**	**0,6901**	-0,0101	**0,400/0,405**	**0,1093**
Crdt-g	69,70 (0,4922)	103/140	0,1193	69,50 (0,5523)	**82/114**	**0,1949**	-0,0029	**0,204/0,186**	**0,6337**
Diab.	74,09 (0,4356)	22/43	0,2993	73,96 (0,5103)	**21/41**	**0,3766**	-0,0018	**0,045/0,047**	**0,2583**
Glass	67,29 (0,2837)	30/59	1,3307	**68,69** (0,2991)	**23/45**	**1,3524**	**0,0208**	**0,233/0,237**	**0,0163**
Heart-c	79,21 (0,2619)	30/51	0,5354	**79,21** (0,2884)	**18/29**	**0,5917**	**0,0000**	**0,400/0,431**	**0,1052**
Heart-s	77,78 (0,4322)	18/35	0,4847	75,93 (0,4907)	21/41	**0,5054**	-0,0238	-0,167/-0,171	**0,0427**
Hepat.	79,35 (0,4200)	11/21	0,1445	77,42 (0,4752)	**8/15**	**0,1510**	-0,0244	**0,273/0,286**	**0,0450**
Ionos.	90,88 (0,2887)	18/35	0,7416	89,74 (0,3203)	**16/31**	0,7247	-0,0125	**0,111/0,114**	-0,0228
Iris	95,33 (0,1707)	5/9	1,4663	94,67 (0,1886)	**3/5**	**1,4692**	-0,0070	**0,400/0,444**	**0,0020**
Labor	78,95 (0,4285)	3/5	0,3496	**80,70** (0,4393)	3/5	**0,5257**	**0,0222**	**0,000/0,000**	**0,5037**
Lymph	77,03 (0,3274)	21/34	0,6074	**77,70** (0,3339)	**18/28**	**0,6712**	**0,0088**	**0,143/0,176**	**0,1050**
Prim.t.	40,71 (0,1961)	47/88	1,2050	**41,30** (0,2310)	**29/54**	**1,2396**	**0,0145**	**0,383/0,386**	**0,0287**
Segment	97,14 (0,0893)	39/77	2,7130	96,02 (0,1067)	52/103	2,6867	-0,0116	-0,333/-0,338	-0,0097
Sonar	74,04 (0,4986)	18/35	0,4662	**75,81** (0,4952)	**15/29**	**0,5050**	**0,0239**	**0,167/0,171**	**0,0832**
T-tac-t	84,76 (0,3485)	95/142	0,5613	78,91 (0,4592)	**71/106**	0,4793	-0,0690	**0,253/0,254**	-0,1461
Vehicle	73,40 (0,3272)	98/195	1,3607	70,92 (0,3813)	**56/111**	1,2996	-0,0338	**0,429/0,431**	-0,0449
Vote	96,78 (0,1650)	6/11	0,8580	96,55 (0,1857)	6/11	**0,8908**	-0,0024	**0,000/0,000**	**0,0382**
Wine	94,94 (0,1776)	5/9	1,4476	93,26 (0,2120)	7/13	1,4192	-0,0178	-0,400/-0,444	-0,0196
Zoo	92,08 (0,1359)	9/17	2,1435	**92,08** (0,1504)	9/17	2,1014	**0,0000**	**0,000/0,000**	-0,0196
Average	82,10 (0,2900)	34,71/ 58,32	1,01	80,38 (0,3300)	**24,39/ 41,54**	1,01	-0,0195	**0,037/ 0,051**	**0,0960**

Results of experiments show that there is no significant difference in average accuracy between the consensus decision trees and the decision trees induced by C4.5 ($t = 1.8664$, $df = 27$, $p = 0.0729$, using two-tailed t-test for dependent samples, where t, df and p stand for *t-statistics*, *degrees of freedom* and *significance level*, respectively), using a 95% significance level (the bound used throughout this paper). Notice, however, that the CDT algorithm improves the information score (compared to C4.5) in 18 domains (9.6% improvement on the average).

Our hypothesis that the structure of CDT is simpler than the structure of the induced C4.5 decision trees was confirmed: indeed, the average number of leaves of CDT is significantly smaller than the average number of leaves of the C4.5 decision trees ($t = 2.3787$, $df = 27$, $p = 0.0247$, using two-tailed t-test for dependent samples). Moreover, the average tree size (measured by the number

of all decision tree nodes) of CDT is also significantly smaller than that of C4.5 ($t = 2.4413$, $df = 27$, $p = 0.0215$, using two-tailed t-test for dependent samples).

The relative improvement in tree size also shows that in 19 domains the CDT algorithm learned smaller decision trees than C4.5 yielding, on the average, 3.7% smaller trees according to the number of leaves (5.1% according to the number of all nodes).

5 Summary and Conclusions

Results show that consensus decision trees are, on the average, as accurate as C4.5 decision trees, but simpler (smaller w.r.t. the number of leaves and nodes). Moreover, consensus decision trees improve the information score compared to C4.5 decision trees[4].

We also tested alternative ways of constructing consensus decision trees. First, the similarity measure that only considers the distances between instances was replaced by a measure that takes into the account that instances are labelled by class labels; the similarity value of two instances labelled by the same class label was appropriately increased. Opposed to our expectations, this way of measuring similarities between instances has not proved to be better than the one described in this paper.

Second, instead of labelling instances by class labels, instances may be labelled by cluster labels, considering clusters generated by consensus clustering as classes for learning by C4.5. This approach has turned out to be inferior compared to the approaches described in the paper.

In further work we are planning to measure the similarities between instances not just by counting how many times two instances belong to the same leaf (have the same attribute-value representation), but also by putting different weights on the segments of this path (higher weights assigned to segments closer to the root).

Moreover, our plan is to test the hypothesis that consensus decision trees are more robust with respect to adding of new instances, i.e., that the structure of the consensus decision tree would change less than the structure of C4.5 decision trees. To this end we need to propose new measures of tree structure variability, and measure the robustness accordingly. The current results indicate that a step in the direction of improving the robustness has been achieved, assuming that simpler tree structures are more robust.

There is however a performance drawback that we should take into account when using the CDT method for building decision trees. Since the CDT algorithm builds 11 decision trees (10 in the cross-validation process and the final one) and does also the hierarchical clustering, it is much slower than the traditional decision tree building algorithm.

Acknowledgements. Thanks to Saso Džeroski, Ljupčo Todorovski and Marko Grobelnik for useful comments on the draft of this paper. Thanks also to Bernard

[4] There are two data sets in which a C4.5 decision tree outperforms CDT in accuracy, simplicity and information score; on the other hand, in five domains CDT is better in all the three characteristics.

Ženko for his help when using WEKA. The work reported in this paper was supported by the Slovenian Ministry of Education, Science and Sport, and the IST-1999-11495 project Data Mining and Decision Support for Business Competitiveness: A European Virtual Enterprise.

References

1. Adams, E.N. (1972). Consensus techniques and the comparison of taxonomic trees. *Systematic Zoology*, 21, 390–397.
2. Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA.
3. Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24:123–140,1996.
4. Day, W.H.E. (1983). The role of complexity in comparing classifications. *Mathematical Biosciences*, 66, 97–114.
5. Efron, B. (1979). Bootstrap methods: Another look at the jackknife. *Annals of Statistic*, 7(1): 1–26.
6. Faith, D.P. (1988). Consensus applications in the biological sciences. In: Bock, H.H. (Ed.) *Classification and Related Methods of Data Analysis*, Amsterdam: North-Holland, 325–332.
7. Fisher, D.H. (1989). Noise-tolerant conceptual clustering. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* 825–830. San Francisco: Morgan Kaufmann.
8. Gordon, A.D. (1981). *Classification*. London: Chapman and Hall.
9. Hartigan, J.A. (1975). *Cluster Algorithms*. New York: Wiley.
10. Kohavi, R. (1995). Wrappers for performance enhancement and oblivious decision graphs. Doctoral dissertation, Stanford University.
11. Kononenko, I. and Bratko, I. (1991). Information based evaluation criterion for classifier's performance. *Machine Learning*, 6, (1), 67–80.
12. Langley, P. (1996). *Elements of Machine Learning*. Morgan Kaufmann.
13. Leclerc, B. (1988). Consensus applications in the social sciences. In: Bock, H.H. (Ed.) *Classification and Related Methods of Data Analysis*, Amsterdam: North-Holland, 333–340.
14. McMorris, F.R. and Neuman, D. (1983). Consensus functions defined on trees. *Mathematical Social Sciences*, 4, 131–136.
15. Quinlan, J.R. (1986). Induction of decision trees. *Machine Learning*, 1(1): 81–106.
16. Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. California: Morgan Kaufmann.
17. Regnier, S. (1965). Sur quelques aspects mathematiques des problems de classification automatique. *I.I.C. Bulletin*, 4, 175–191.
18. Scheffer, T. and Herbrich, R. (1997). Unbiased assessment of learning algorithms. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 798–803.
19. Sokal, R.R. and Sneath, P.H.A. (1963). *Principles of Numerical Taxonomy*. San Francisco: Freeman.
20. Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society*, B 36, 111–147.
21. Witten, I.H. and Frank, E. (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco.
22. Zhang, J. (1992). On the distributional properties of model selection criteria. *Journal of the American Statistical Association*, 87(419) 732–737.